If you followed the quickstartguide you now have a configured environment ready to test your first Action.

Make a new Action, implementing net.sourceforge.javajax.JavajaxAction interface and place it in the action package specified in web.xml.

Since your class will be something like my.very.nested.package.MyOldFashionActionClass, and since the URL to call this class depends on the class name, maybe we want to bind the class to a smarter name. Here the UrlBinding annotation come in hands.

Place it before the class declaration: @UrlBinding("MyAction").

All the "callable" methods in the action MUST be "protected", this way we can put in a first method like this:

```
protected Response helloUser(){
...
}
```

Since we want the method to greet the caller, we must put an input parameter:

```
protected Response helloUser(Param("name") String userName){
...
}
```

The annotation Param will inform the framework to bind the request parameter "name" to the method parameter "userName". Since we want to force the user to tell us his/her name, we can make it mandatory and inform the framework what to do when a validation error occurs:

@OnError("myPage.jsp") // return to the same page

```
protected Response helloUser(Param(value="name", mandatory=true) String userName){
...
}
```

Now we have all the information to greet the user:

```
@OnError("myPage.jsp") // return to the same page
protected Response helloUser(@Param(value="name", mandatory=true) String userName){

 return new StringResponse("Hello " + userName);
}
```

Now it is time to call this method. Make a new jsp page, create a form with a text input and two buttons: on to call it with ajax, one to submit the form.

```
 <form name="hello" action="MyAction.helloUser.page">

Name: <input type="text" name="name">



<input type="button" value="Ajax call" onclick="makeCall(this)">



<input type="submit" value="Page call">



</form>
```

for the Ajax part we need some javascript to make the call, so put the function in the section:

```
<script>

 makeCall = function(formElement) {
 var url ="MyAction.helloUser.ajax";
```

```
 var caller = new AjaxCaller(
 formElement,
 {
 onSuccess: function(transport){ alert(transport.responseText); }
 });

 caller.send(url);
};
```

```
</script>
```

Don't forget to include Prototype.js and javajax.js before this script.

For error displaying, we can use the tag

```
<javajax:errors/>
```

For error field highlighting we can use the jsp tag

```
<javajax:highligther form="hello"/>
```

The framework will assign the class "errorfield" to the fields with validation errors, so just make a style definition for that class

```
<style>
```

```
 .errorfield {

 border:1px solid red;

 background:#FFEEAA;

}

</style>
```

Now its time to push those buttons!

Download the source code for this example.